

Francisco Samayoa

Adam Tindale

Atelier I: Discovery-001

September 28, 2019

## Code & Its Many States

<https://github.com/avongarde/Atelier/tree/master/Assignment%201>

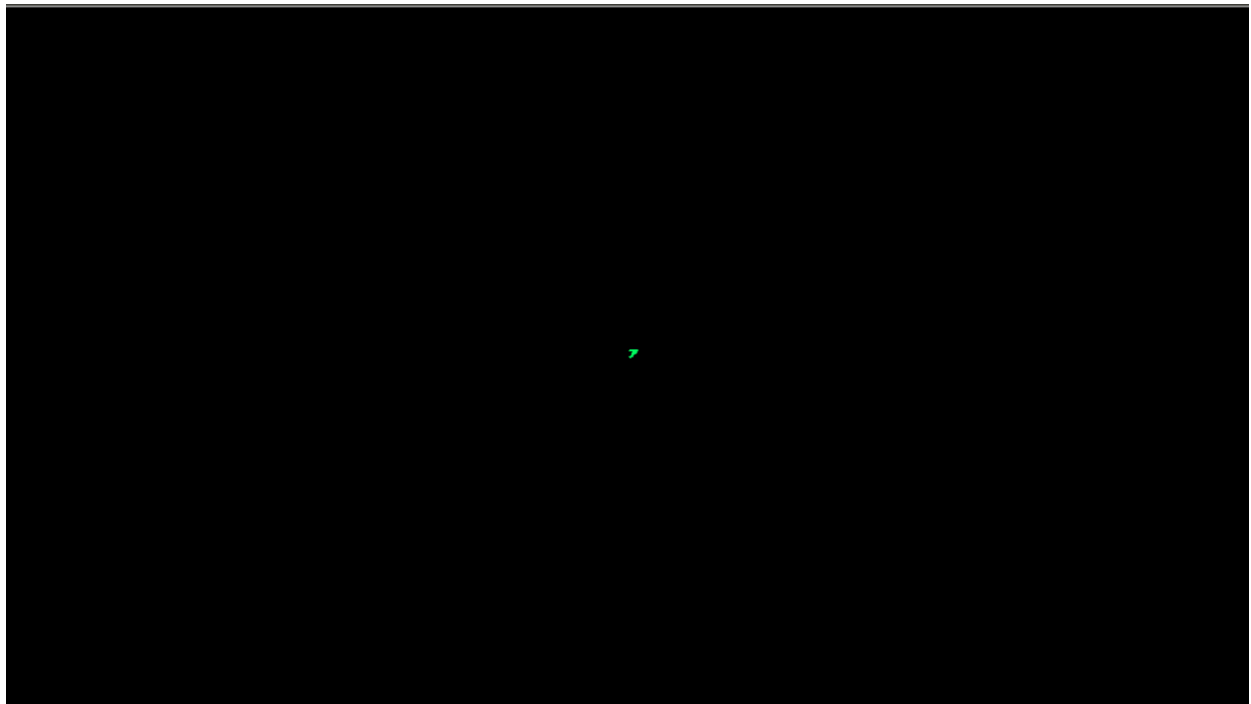
For this project, I wanted to emulate the digital rain from the Matrix. The Matrix is one of my favourite movies, and the iconography associated with it is the falling green code. The code is a way of representing the virtual activity inside the Matrix – a simulated world – on screen. For my interface, I used the *p5.js* programming language and recreated the same effect with my own spin on it. I wanted to show that the user could directly affect the code and its state. The falling green code is associated with uniformity and equilibrium. However, if the mouse veers off to the right of the screen, the code turns red and begins wandering off in many directions. Thus, symbolizing the code's corruption. In conclusion, I used the idea of digital rain and turned it into a visual association of a programmer (the user) encountering code with/without errors. P5.js was perhaps the best language for this project because of my previous knowledge of JavaScript and Processing. P5.js is essentially a sketchbook for your browser and is accessible for artists and designers.

My father and I are movie buffs, and the Matrix has stood the test of time as one of our shared picks. I've always linked the movie to computer programming, and now that I'm learning it in school I wanted to explore the possibility of emulating something from it. When I began to learn p5, I instantly sought out ways to recreate the digital rain. I came across a tutorial on YouTube that showed you exactly how to do it. Actually, it was on *The Coding Train*, a channel

I am quite familiar with. But before that, I wanted to create as much of it as I could on my own. I ended up creating the Symbol class on my own, using the `String.fromCharCode()` method and displayed one symbol on the screen. From there I populated the screen with symbols using an array, with much dissatisfaction. After that, it became increasingly more difficult, even with my knowledge and some help from a friend. I ended up referencing the tutorial but – including the code corruption aspect – most the final code is original. One aspect I wish I improved on was making the canvas full screen.

<https://www.youtube.com/watch?v=S1TQCi9axzg>

[https://www.w3schools.com/jsref/jsref\\_fromcharcode.asp](https://www.w3schools.com/jsref/jsref_fromcharcode.asp)



*Figure 1. One symbol centered on the canvas*

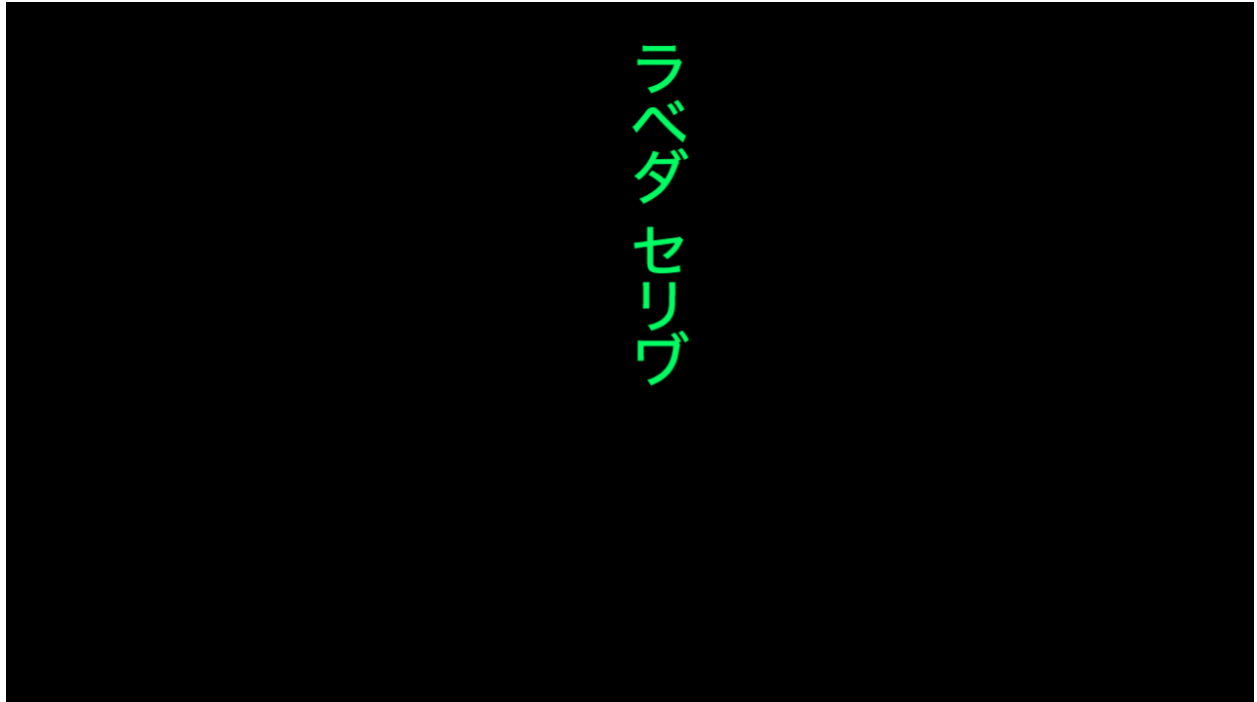


Figure 2. A stream of symbols cantered on the canvas falling from above

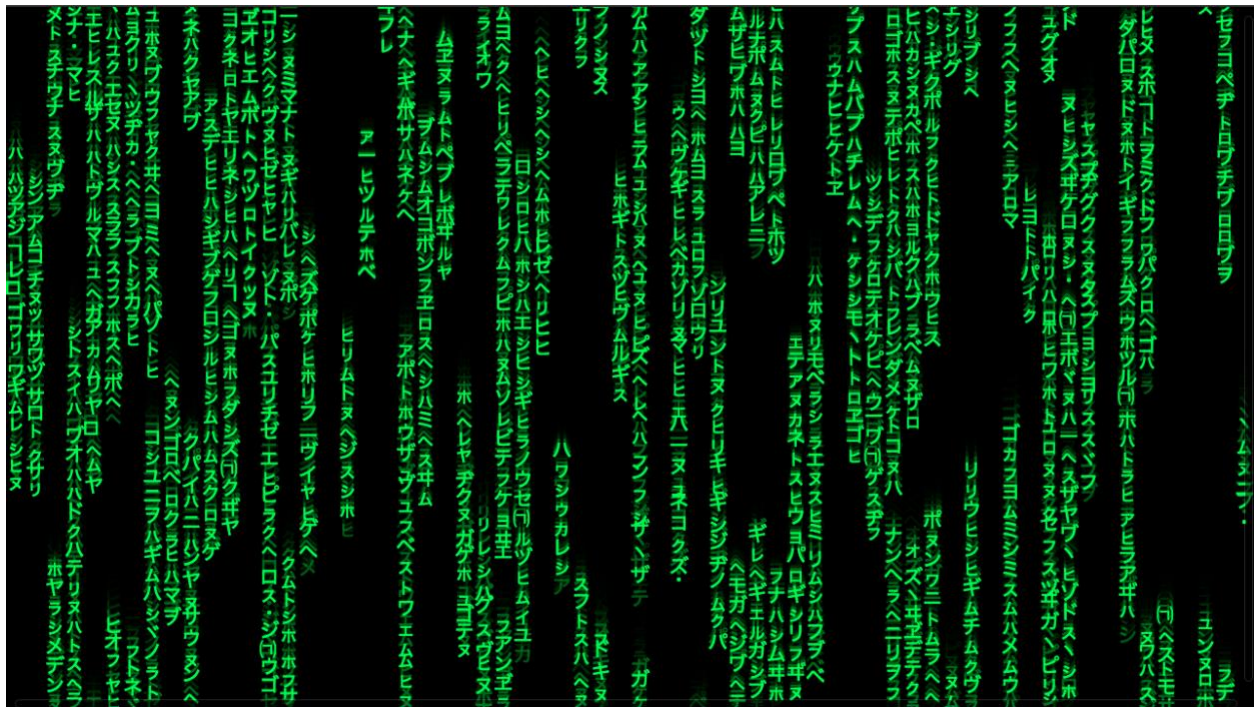


Figure 3. Version 1.0: Green symbols populating the screen and falling from above



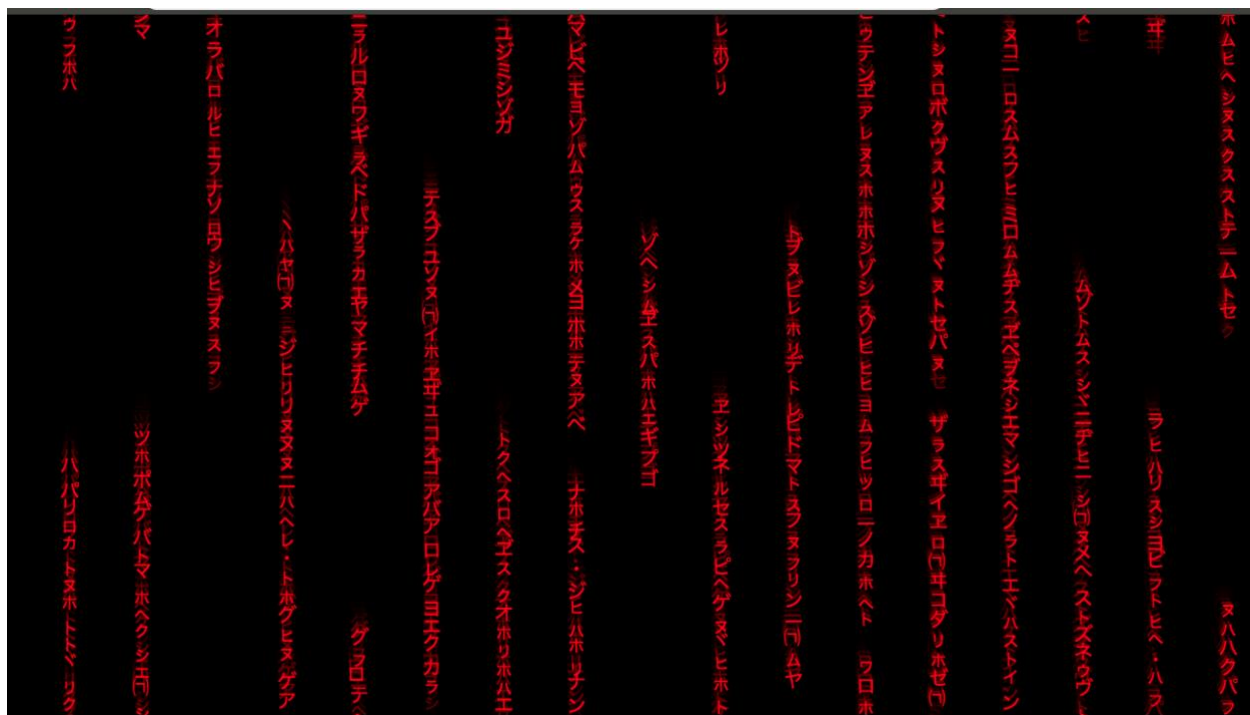


Figure 4abc. Version 2.5: MouseX is incorporated; affects the fill colour and symbols' x-value